Commodification of Accelerations for the Karp and Miller Construction.

Alain Finkel · Serge Haddad · Igor Khmelnitsky

Abstract Karp and Miller's algorithm is based on an exploration of the reachability tree of a Petri net where, the sequences of transitions with positive incidence are accelerated. The tree nodes of Karp and Miller are labeled with ω-markings representing (potentially infinite) coverability sets. This set of ω-markings allows us to decide several properties of the Petri net, such as whether a marking is coverable or whether the reachability set is finite. The edges of the Karp and Miller tree are labeled by transitions but the associated semantic is unclear which yields to a complex proof of the algorithm correctness. In this work we introduce three concepts: abstraction, acceleration and exploration sequence. In particular, we generalize the definition of transitions to ω-transitions in order to represent accelerations by such transitions. The notion of abstraction makes it possible to greatly simplify the proof of the correctness. On the other hand, for an additional cost in memory, which we theoretically evaluated, we propose an "accelerated" variant of the Karp and Miller algorithm with an expected gain in execution time. Based on a similar idea we have accelerated (and made complete) the minimal coverability graph construction, implemented it in a tool and performed numerous promising benchmarks issued from realistic case studies and from a random generator of Petri nets.

Keywords Petri nets · Coverability · Verification tool

A. Finkel
ENS Paris-Saclay, LSV, Université Paris-Saclay, Cachan, France
E-mail: finkel@lsv.fr

S. Haddad
ENS Paris-Saclay, LSV, Université Paris-Saclay, INRIA Cachan, France
E-mail: haddad@lsv.fr

I. Khmelnitsky
ENS Paris-Saclay, LSV, Université Paris-Saclay, INRIA Cachan, France
E-mail: khmelnitsky@lsv.fr

1 Introduction

Coverability and Karp and Miller's algorithm. The coverability set (also denoted as *Cover*) of a Petri net with an initial marking is the downward closure (for the usual order on integer vectors) of the set of reachable markings. An effective finite representation of the cover makes it possible to decide several problems such as: Can a given marking be covered by an reachable marking (the coverability problem)? Is the set of reachable markings finite? Which places are unbounded?

In 1969, Karp and Miller showed that a finite representation of the coverability set of Petri nets and vector addition systems is computable by an algorithm constructing a finite tree (KMT) [13] whose finite set of vertex labels (ω-markings) C represents the cover. Specifically, the downward closure (in $\mathbb{N}^P$) of C, denoted $\downarrow C$, coincides with the cover. The set C is not unique because it depends on the order chosen for the exploration of the successors of the tree nodes. Moreover, it is not minimal in the number of elements because it may can contain comparable ω-markings and thus contain redundant items.

The original proof of Karp and Miller's algorithm is incomplete as Hack had already noted in 1974 [11]. Moreover the proofs of variants of the Karp and Miller algorithm (see below) are difficult and do not reuse the original proof of Karp and Miller. Motivated by the lack of complete and certified proof, Yamamoto et al wrote a formal COQ proof of the correctness of Karp and Miller's algorithm [19].

Clover, the canonical representation of the cover. It is possible to associate with any marked Petri net a finite and *canonical* representation of the cover. Indeed, any downward closed set in $\mathbb{N}^P$ is equal to the downward closed set (in $\mathbb{N}^P$) consisting of a finite subset of incomparable ω-markings in $\mathbb{N}_\omega^P$. Thus one can associate with any marked Petri net a *unique* finite representation of its cover [6]. This approach is generalized to monotonic transition systems with a well quasi order (i.e. well-structured systems) [5] and even to monotonic transition systems with an order without infinite antichains [2]. This finite representation is called *Clover* (for *Clo*sure of the C*over*) in [8]. This set is minimal, unique, and consists of maximal elements (these elements represent particular downward closed sets, called ideals). The Clover can be computed from C, by keeping only the maximal elements. Once the Clover is obtained, one can answer coverability questions without rerunning the coverability algorithm each time since one only needs to compare the desired marking to be covered with the markings in the Clover, which takes time which is proportional to the size of Clover and no longer necessarily doubly exponential. Clover also makes it possible to answer a wider variety of questions beyond coverability and finiteness problems. Let us illustrate this point with the following question on ω-coverability (which by itself represents an infinity number of coverability questions): Is the marking $(n, 2, 5, n)$ coverable for all $n \geq 0$? This property holds if and only if the ω-marking $(\omega, 2, 5, \omega)$ is smaller than another ω-marking in the clover, which can be tested in time proportional to the size of Clover.

The Clover set thus allows to solve many problems without re-running an algorithm which is doubly exponential. The question which arises is whether one can find an efficient algorithms to compute the Clover of a marked Petri net.

Variants of the Karp and Miller algorithm. In [6] the author develops an a modification to the Karp and Miller algorithm where at any time during the

execution of the algorithm, the current set of labeled vertices form an antichain (a set of incomparable elements). The algorithm consists of accelerating, like the original algorithm, the current node and then *removing* all the sub-trees whose marked root is strictly covered by the marking of the current node. Conversely, if the marking of the current node is covered by the marking of an existing node then the exploration from the current node is stopped and it is removed from the tree. Unfortunately this algorithm contains a bug, identified in 2005, and for some executions calculates a strict sub-approximation of Clover [7]. Since 2005, three main algorithms (with variants) have been proposed [10,17,18,16] to calculate Clover without completely building the Karp and Miller tree. An empirical evaluation of these three algorithms gave rather good performances on most of the commonly analyzed case studies but no theoretical limit of the additional cost in memory of these algorithms compared to the algorithm developed in [6] is known.

Our contribution. First we give a simple and elegant proof of the Karp and Miller algorithm based on three new concepts: *abstraction*, *acceleration* and *sequence of exploration*. In particular we transform the accelerations of the Karp and Miller algorithm to first-class citizens (i.e. commodification of the accelerations) instead of using them implicitly. Then we propose two "accelerated" variants of the Karp and Miller algorithm with an expected gain in execution time. Based on a similar idea we have accelerated (and made complete) the minimal coverability graph construction of Alain Finkel [6], implemented it in a tool and performed numerous promising benchmarks issued from realistic case studies and from a random generator of Petri nets.

- An abstraction is an ω-transition (i.e. a generalized transition), where (1) its backward incidence and incidence with respect to a place can be equal to ω (i.e. belonging to $\mathbb{N}_\omega$) and (2) which has an infinite family of transition sequences "justifying" the introduction of the ω's. We show that their firing from a ω-marking whose associated ideal is included in Cover leads to a ω-marking whose associated ideal is also included in Cover. We then prove that the concatenation of abstractions is still an abstraction. An acceleration is an abstraction whose incidence with respect to each place is either zero or ω. We establish that any abstraction can be transformed into an acceleration by substituting the strictly positive components of the incidence by ω and requiring ω tokens for the strictly negative components of the incidence.
- The proof of the Karp and Miller algorithm becomes rather simple, with the addition of ghost variables (i.e. variables without effect on the execution of the algorithm). The proof of the termination is based on the well order of $\mathbb{N}_\omega^P$. The proof of consistency is an almost immediate consequence of the properties of abstractions and accelerations. The proof of completeness is based on the notion of exploration sequences detailed below.
- We then deepen our study of accelerations. The set of accelerations provided with a natural order is a well order, i.e. it can be represented by its finite base of minimal elements. We show that the integer coefficients of the minimum accelerations are bounded by $B(e, d)$, which is polynomial in the size of the incidence matrices e and doubly exponential in the number of places d. We also show how to transform (*truncate*) any acceleration into an acceleration whose integer coefficients are bounded by $B(e, d)$. We then propose an accelerated version of the algorithm of Karp and Miller with an expected gains in

the execution time. The general principle is as follows: when you discover an acceleration, you truncate it and memorize it. Then at each step of the algorithm, the marking of the current node is increased by firing the accelerations that can be fired. Due to the truncation of the accelerations, our accelerated version of the algorithm of Karp and Miller requires a minimal additional cost in memory, compared to the general cost memory of the algorithm of Karp and Miller which is non-primitive recursive. In addition, the proof of the correctness of our accelerated variant is immediately deduced from our original proof.
- A second version consists to memorize accelerations and reuse it along the incomplete algorithm of [6]. We proved that this version recovers completeness in [9]. Here we perform a large empirical study of the efficiency of this approach that we have integrated in our tool `MinCov`.

Organization. In Section 2 we introduce and study abstractions and accelerations of a Petri net. We then establish the proof of the Karp and Miller algorithm in Section 3. In Section 4 we describe our version of the Karp and Miller algorithm. In Section 5 we describe and evaluate our accelerated version of the algorithm of [6]. Finally we conclude and offer perspectives to our work in Section 6.

2 Covering and Abstractions

2.1 Petri nets: reachability and coverability

We define the Petri nets here in a different but equivalent way compared to the usual definition, namely using the backward incidence matrix $\mathbf{Pre}$ and the incidence matrix $\mathbf{C}$. The forward incidence matrix is implicitly defined by $\mathbf{C}+\mathbf{Pre}$. This choice is justified by the introduction of abstractions in Subsection 2.2.

Definition 1 A Petri net (PN) is a tuple $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{C}\rangle$ where:

- P is a finite set of *places*;
- T is a finite set of *transitions* with $P \cap T = \emptyset$;
- $\mathbf{Pre} \in \mathbb{N}^{P\times T}$ is the *backward incidence matrix*;
- $\mathbf{C} \in \mathbb{Z}^{P\times T}$ is the *incidence matrix* for which the following holds: For all $p \in P$ and $t \in T$, $\mathbf{C}(p,t) + \mathbf{Pre}(p,t) \geq 0$.

A marked PN $(\mathcal{N}, \mathbf{m}_0)$ is a PN $\mathcal{N}$ initialized with a *marking* $\mathbf{m}_0 \in \mathbb{N}^P$.

The column vector of the matrix $\mathbf{Pre}$ (resp. $\mathbf{C}$) indexed by $t \in T$ is denoted by $\mathbf{Pre}(t)$ (resp. $\mathbf{C}(t)$). A transition $t \in T$ is *fireable* from a marking $\mathbf{m} \in \mathbb{N}^P$ if $\mathbf{m} \geq \mathbf{Pre}(t)$. When t is fired from a marking $\mathbf{m}$, its *firing* leads to a marking $\mathbf{m}' \stackrel{\text{def}}{=} \mathbf{m} + \mathbf{C}(t)$, which is denoted by $\mathbf{m} \xrightarrow{t} \mathbf{m}'$. We extend the firing rule to a sequence of firings $\sigma \in T^*$ recursively according to its length, as follows: The empty sequence ε is always fireable and does not change the marking. The sequence $\sigma = t\sigma'$, with $t \in T$ and $\sigma' \in T^*$ is fireable from $\mathbf{m}$ if $\mathbf{m} \xrightarrow{t} \mathbf{m}'$ and σ' is fireable from $\mathbf{m}'$. The firing of σ from $\mathbf{m}$ leads to a marking $\mathbf{m}''$ reached by σ' from $\mathbf{m}'$. We denote this firing by $\mathbf{m} \xrightarrow{\sigma} \mathbf{m}''$.

Definition 2 Let $(\mathcal{N}, \mathbf{m}_0)$ be a marked PN. The *reachability set* $Reach(\mathcal{N}, \mathbf{m}_0)$ is defined by:

$$Reach(\mathcal{N}, \mathbf{m}_0) = \{\mathbf{m} \mid \exists \sigma \in T^* \ \mathbf{m}_0 \xrightarrow{\sigma} \mathbf{m}\}$$

Before introducing the coverability set of a marked PN, we recall some facts and definitions related to ordered sets. Let $(X, \leq)$ be an ordered set, we will just call X an ordered set when the order is implicit. The downward (resp. upward) *closure* of a set $E \subseteq X$ denoted by $\downarrow E$ (resp. $\uparrow E$) is defined by:

$$\downarrow E = \{x \in X \mid \exists y \in E \ y \geq x\} \qquad (\text{resp. } \uparrow E = \{x \in X \mid \exists y \in E \ y \leq x\})$$

A set $E \subseteq X$ is said to be *downward closed* (resp. *upward closed*) if $E = \downarrow E$ (resp. $E = \uparrow E$). An *antichain* E is a set for which : $\forall x \neq y \in E \ \neg(x \leq y \vee y \leq x)$. X is *FAC* if all of its antichains are finite. A set $E \subset X$ is *directed* if E is nonempty and for all $x, y \in E$ there exists $z \in E$ such that $x \leq z$ and $y \leq z$. An *ideal* is a directed downward closed set. A well known characterization of FAC sets is: a set is FAC if and only if it is equal to a finite union of ideals (one can find a proof of this result in [2]). Given a set $E \subseteq X$, there may exist several finite families of ideals whose union is equal to E. Among all these finite families, one can choose *the unique* set of *maximal* ideals (by inclusion): this set is therefore canonically associated with E.

Recall that an ordered set $(X, \leq)$ is *well founded* if all strictly decreasing sequences are finite and $(X, \leq)$ is *well ordered* if it is well founded and FAC. Another characterization of well ordered sets is: a set $(X, \leq)$ is well ordered if and only if for all sequences $(x_n)_{n \in \mathbb{N}}$ of elements of X, there exists an infinite non decreasing subsequence. Finally recall that $(\mathbb{N}, \leq)$ and $(\mathbb{N}^P, \leq)$ are well ordered.

We are now able to introduce the *cover* (also called the coverability set) of a net and to study some of its properties.

Definition 3 Let $(\mathcal{N}, \mathbf{m}_0)$ be a marked PN. The *coverability set* $Cover(\mathcal{N}, \mathbf{m}_0)$ is defined by:

$$Cover(\mathcal{N}, \mathbf{m}_0) = \downarrow Reach(\mathcal{N}, \mathbf{m}_0)$$

Since the coverability set is downward closed and $\mathbb{N}^P$ is FAC, it can be expressed as a finite union of ideals. The ideals of $\mathbb{N}^P$ can be elegantly defined as follows. We first extend the natural numbers and integers: $\mathbb{N}_\omega = \mathbb{N} \cup \{\omega\}$ and $\mathbb{Z}_\omega = \mathbb{Z} \cup \{\omega\}$. Then we extend the order relation and the addition operation to $\mathbb{Z}_\omega$: For all $n \in \mathbb{Z}$, $\omega > n$ and for all $n \in \mathbb{Z}_\omega$, $n + \omega = \omega + n = \omega$. $\mathbb{N}_\omega^P$ with this extended order is still well ordered and its elements are called *ω-markings*. There is a one to one correspondence between ideals of $\mathbb{N}^P$ and ω-markings. Let $\mathbf{m} \in \mathbb{N}_\omega^P$. Denote by $[\![\mathbf{m}]\!]$ the set:

$$[\![\mathbf{m}]\!] = \{\mathbf{m}' \in \mathbb{N}^P \mid \mathbf{m}' \leq \mathbf{m}\}$$

$[\![\mathbf{m}]\!]$ is an ideal of $\mathbb{N}^P$ (and any ideal can be represented as such). By the definitions and properties stated above, we are able to formally define the Clover of a Petri net.

Definition 4 Let $(\mathcal{N}, \mathbf{m}_0)$ be a marked PN. Then $Clover(\mathcal{N}, \mathbf{m}_0) \subseteq \mathbb{N}_\omega^P$ is the set of maximal ideals such that :

$$Cover(\mathcal{N}, \mathbf{m}_0) = \bigcup_{\mathbf{m} \in Clover(\mathcal{N}, \mathbf{m}_0)} [\![\mathbf{m}]\!]$$

Remark: We can show that the $Clover(\mathcal{N}, \mathbf{m}_0)$ is thus well defined in a unique way and that it is the *smallest* (in number of elements) finite set among all the

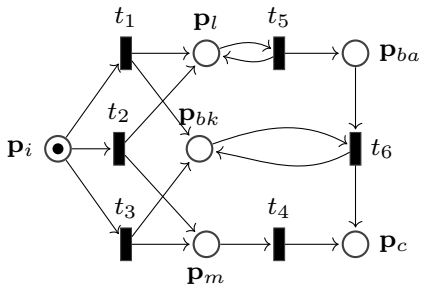

Fig. 1 An unbounded Petri net

finite sets of ideals whose union is equal to the $Cover(\mathcal{N}, \mathbf{m}_0)$.

In [2] one can find a more general definition of a *Clover* for well structured transition systems. One of the goals of the Karp and Miller algorithm is the computation of $Clover(\mathcal{N}, \mathbf{m}_0)$.

Example 1 The PN in Figure 1 is unbounded. Its *Clover* is a set of four elements:

$$\{p_i, p_{bk} + p_m, p_l + p_m + \omega p_{ba}, p_l + p_{bk} + \omega p_{ba} + \omega p_c\}$$

For example, the marking $p_l + p_{bk} + \alpha p_{ba} + \beta p_c$ is reachable by the sequence $t_1 t_5^{\alpha+\beta} t_6^{\beta}$ and therefore covered.

2.2 Abstractions and Accelerations

In order to introduce abstractions and accelerations, we generalize the transitions to take into account place markings with ω tokens.

Definition 5 Let P be a set of places. An ω-transition $\mathbf{a}$ is defined by:

- $\mathbf{Pre}(\mathbf{a}) \in \mathbb{N}_\omega^P$ its *backward incidence*;
- $\mathbf{C}(\mathbf{a}) \in \mathbb{Z}_\omega^P$ its *incidence* with $\mathbf{Pre}(\mathbf{a}) + \mathbf{C}(\mathbf{a}) \geq 0$.

For simplicity, we denote $\mathbf{Pre}(\mathbf{a})(p)$ (resp. $\mathbf{C}(\mathbf{a})(p)$) by $\mathbf{Pre}(p, \mathbf{a})$ (resp. $\mathbf{C}(p, \mathbf{a})$). An ω-transition $\mathbf{a}$ is fireable from an ω-marking $\mathbf{m} \in \mathbb{N}_\omega^P$ if $\mathbf{m} \geq \mathbf{Pre}(\mathbf{a})$. When $\mathbf{a}$ is fired from an ω-marking $\mathbf{m}$, it leads to an ω-marking $\mathbf{m}' \stackrel{\text{def}}{=} \mathbf{m} + \mathbf{C}(\mathbf{a})$, which we denote $\mathbf{m} \xrightarrow{\mathbf{a}} \mathbf{m}'$. Note that if $\mathbf{Pre}(p, \mathbf{a}) = \omega$ then whatever the the value of $\mathbf{C}(p, \mathbf{a})$ is, one has $\mathbf{m}'(p) = \omega$. So without loss of generality, we suppose that for all ω-transition $\mathbf{a}$, $\mathbf{Pre}(p, \mathbf{a}) = \omega$ implies $\mathbf{C}(p, \mathbf{a}) = \omega$.

In order to define abstractions, we define the incidence of a sequence of ω-transitions σ by recurrence over its length. Like previously, we introduce $\mathbf{Pre}(p, \sigma) \stackrel{\text{def}}{=} \mathbf{Pre}(\sigma)(p)$ and $\mathbf{C}(p, \sigma) \stackrel{\text{def}}{=} \mathbf{C}(\sigma)(p)$. The base case corresponds to the definition of an ω-transition. Let $\sigma = t\sigma'$, with t an ω-transition and σ' a sequence of ω-transitions, then:

- $\mathbf{C}(\sigma) = \mathbf{C}(t) + \mathbf{C}(\sigma')$;
- For all $p \in P$
  - if $\mathbf{C}(p,t) = \omega$ then $\mathbf{Pre}(p,\sigma) = \mathbf{Pre}(p,t)$;
  - else $\mathbf{Pre}(p,\sigma) = \max(\mathbf{Pre}(p,t), \mathbf{Pre}(p,\sigma') - \mathbf{C}(p,t))$.

The sequence σ is fireable from $\mathbf{m}$ if and only if $\mathbf{m} \geq \mathbf{Pre}(\sigma)$. In this case, $\mathbf{m} \xrightarrow{\sigma} \mathbf{m} + \mathbf{C}(\sigma)$.

An *abstraction* of a PN is an ω-transition which concisely reflects the behavior of the net from the point of view of coverability (see Proposition 1). We note that a transition t of a PN is by construction (with $\sigma_n = t$) an abstraction.

Definition 6 Let $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{C}\rangle$ be a PN and $\mathbf{a}$ be an ω-transition. $\mathbf{a}$ is an *abstraction* if for all $n \geq 0$, there exists $\sigma_n \in T^*$ such that for all $p \in P$ with $\mathbf{Pre}(p,\mathbf{a}) \in \mathbb{N}$:

1. $\mathbf{Pre}(p,\sigma_n) \leq \mathbf{Pre}(p,\mathbf{a})$;
2. if $\mathbf{C}(p,\mathbf{a}) \in \mathbb{Z}$ then $\mathbf{C}(p,\sigma_n) \geq \mathbf{C}(p,\mathbf{a})$;
3. if $\mathbf{C}(p,\mathbf{a}) = \omega$ then $\mathbf{C}(p,\sigma_n) \geq n$.

The following proposition justifies the interest of abstractions.

Proposition 1 *Let $(\mathcal{N}, \mathbf{m}_0)$ be a marked PN, $\mathbf{a}$ be an abstraction and $\mathbf{m}$ be an ω-marking such that: $[\![\mathbf{m}]\!] \subseteq Cover(\mathcal{N}, \mathbf{m}_0)$ and $\mathbf{m} \xrightarrow{\mathbf{a}} \mathbf{m}'$.*
Then $[\![\mathbf{m}']\!] \subseteq Cover(\mathcal{N}, \mathbf{m}_0)$.

Proof Let $\mathbf{m}^* \in [\![\mathbf{m}']\!]$. Denote $n = \max(\mathbf{m}^*(p) \mid \mathbf{m}'(p) = \omega)$
and $\ell = \max(\mathbf{Pre}(p,\sigma_n), n - \mathbf{C}(p,\sigma_n) \mid \mathbf{m}(p) = \omega)$.
Define $\mathbf{m}^\sharp \in [\![\mathbf{m}]\!]$ by:

- If $\mathbf{m}(p) < \omega$ then $\mathbf{m}^\sharp(p) = \mathbf{m}(p)$;
- Otherwise $\mathbf{m}^\sharp(p) = \ell$.

Let us check that σ_n is fireable from $\mathbf{m}^\sharp$. For any $p \in P$,

- If $\mathbf{m}(p) < \omega$ then $\mathbf{m}^\sharp(p) = \mathbf{m}(p) \geq \mathbf{Pre}(p,\mathbf{a}) \geq \mathbf{Pre}(p,\sigma_n)$;
- Otherwise $\mathbf{m}^\sharp(p) = \ell \geq \mathbf{Pre}(p,\sigma_n)$.

Let us check that $\mathbf{m}^\sharp + \mathbf{C}(\sigma_n) \geq \mathbf{m}^*$. For any $p \in P$,

- If $\mathbf{m}(p) < \omega$ and $\mathbf{C}(p,\mathbf{a}) < \omega$ then
  $\mathbf{m}^\sharp(p) + \mathbf{C}(p,\sigma_n) \geq \mathbf{m}(p) + \mathbf{C}(p,\mathbf{a}) = \mathbf{m}'(p) \geq \mathbf{m}^*(p)$;
- If $\mathbf{m}(p) < \omega$ and $\mathbf{C}(p,\mathbf{a}) = \omega$ then
  $\mathbf{m}^\sharp(p) + \mathbf{C}(p,\sigma_n) \geq \mathbf{C}(p,\sigma_n) \geq n \geq \mathbf{m}^*(p)$;
- If $\mathbf{m}(p) = \omega$ then $\mathbf{m}^\sharp(p) + \mathbf{C}(p,\sigma_n) \geq n - \mathbf{C}(p,\sigma_n) + \mathbf{C}(p,\sigma_n) = n \geq \mathbf{m}^*(p)$.

A simple way to build new abstractions consists in concatenating them.

Proposition 2 *Let $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{C}\rangle$ be a PN and σ a sequence of abstractions. Then the ω-transition $\mathbf{a}$ defined by $\mathbf{Pre}(\mathbf{a}) = \mathbf{Pre}(\sigma)$ and $\mathbf{C}(\mathbf{a}) = \mathbf{C}(\sigma)$ is an abstraction.*

Proof We show this result by recurrence on the length of σ. The base case is immediate. Let $\sigma = b\sigma'$ and (by the recursion hypothesis) let $\{\sigma'_n\}_{n\in\mathbb{N}}$ a family of sequences of transitions associated with σ'. Let $\{\sigma_{n,b}\}_{n\in\mathbb{N}}$ a family of sequences of transitions associated with b. Fix $n \in \mathbb{N}$.
Denote by $n' = \max(n, \max(n - \mathbf{C}(p,b) \mid \mathbf{C}(p,b) < \omega = \mathbf{C}(p,\sigma')))$.
Denote $\ell = \max(\mathbf{Pre}(p,\sigma'_{n'}), n - \mathbf{C}(p,\sigma'_{n'}) \mid \mathbf{Pre}(p,b) < \omega = \mathbf{C}(p,b))$.
Let us show that $\sigma_{\ell,b}\sigma'_{n'}$ satisfies the conditions of Definition 6.
Let $p \in P$, $\mathbf{Pre}(p,\mathbf{a}) < \omega$ if and only if (1) $\mathbf{Pre}(p,b) < \omega$ and $\mathbf{C}(p,b) = \omega$ or (2) $\mathbf{Pre}(p,b) < \omega$, $\mathbf{C}(p,b) < \omega$ and $\mathbf{Pre}(p,\sigma') < \omega$.

• **Case $\mathbf{Pre}(p,b) < \omega$ and $\mathbf{C}(p,b) = \omega$.**
Therefore $\mathbf{Pre}(p,\mathbf{a}) = \mathbf{Pre}(p,b)$ and $\mathbf{C}(p,\mathbf{a}) = \omega$.
We thus have $\mathbf{Pre}(\sigma_{\ell,b}) \leq \mathbf{Pre}(p,b) = \mathbf{Pre}(p,\mathbf{a})$.
Moreover $\mathbf{Pre}(p,\sigma_{\ell,b}) + \mathbf{C}(p,\sigma_{\ell,b}) \geq \mathbf{C}(p,\sigma_{\ell,b}) \geq \ell \geq \mathbf{Pre}(p,\sigma'_{n'})$.
Finally $\mathbf{C}(p,\sigma_{\ell,b}) + \mathbf{C}(p,\sigma'_{n'}) \geq \ell + \mathbf{C}(p,\sigma'_{n'}) \geq n - \mathbf{C}(p,\sigma'_{n'}) + \mathbf{C}(p,\sigma'_{n'}) \geq n$.

• **Case $\mathbf{Pre}(p,b) < \omega$, $\mathbf{C}(p,b) < \omega$ and $\mathbf{Pre}(p,\sigma') < \omega$.**
Therefore $\mathbf{Pre}(p,\mathbf{a}) = \max(\mathbf{Pre}(p,b), \mathbf{Pre}(p,\sigma') - \mathbf{C}(p,b))$ and:

$$\begin{aligned}\mathbf{Pre}(p,\sigma_{\ell,b}\sigma'_{n'}) &= \max(\mathbf{Pre}(p,\sigma_{\ell,b}), \mathbf{Pre}(p,\sigma'_{n'}) - \mathbf{C}(p,\sigma_{\ell,b}))\\ &\leq \max(\mathbf{Pre}(p,b), \mathbf{Pre}(p,\sigma') - \mathbf{C}(p,b))\\ &= \mathbf{Pre}(p,\mathbf{a})\end{aligned}$$

There are now two sub-cases to be considered.

∘ $\mathbf{C}(p,\sigma') < \omega$. So, $\mathbf{C}(p,\mathbf{a}) = \mathbf{C}(p,b) + \mathbf{C}(p,\sigma')$
and $\mathbf{C}(p,\sigma_{\ell,b}\sigma'_n) = \mathbf{C}(p,\sigma_{\ell,b}) + \mathbf{C}(p,\sigma'_n) \geq \mathbf{C}(p,b) + \mathbf{C}(p,\sigma') = \mathbf{C}(p,\mathbf{a})$.

∘ $\mathbf{C}(p,\sigma') = \omega$. So, $\mathbf{C}(p,\mathbf{a}) = \omega$
and $\mathbf{C}(p,\sigma_{\ell,b}\sigma'_{n'}) = \mathbf{C}(p,\sigma_{\ell,b}) + \mathbf{C}(p,\sigma'_{n'}) \geq \mathbf{C}(p,b) + n - \mathbf{C}(p,b) = n$.
Therefore $\mathbf{a}$ is an abstraction.

We now introduce the concept underlying the construction of Karp and Miller.

Definition 7 Let $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{C}\rangle$ be a PN. We say that $\mathbf{a}$ is an *acceleration* if $\mathbf{a}$ is an abstraction such that $\mathbf{C}(\mathbf{a}) \in \{0,\omega\}^P$.

The following proposition provides a way of obtaining acceleration from any abstraction.

Proposition 3 *Let $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{C}\rangle$ be a PN and $\mathbf{a}$ be an abstraction. Define $\mathbf{a}'$ an ω-transition by for all $p \in P$*

- *If $\mathbf{C}(p,\mathbf{a}) < 0$ then $\mathbf{Pre}(p,\mathbf{a}') = \mathbf{C}(p,\mathbf{a}') = \omega$;*
- *If $\mathbf{C}(p,\mathbf{a}) = 0$ then $\mathbf{Pre}(p,\mathbf{a}') = \mathbf{Pre}(p,\mathbf{a})$ and $\mathbf{C}(p,\mathbf{a}') = 0$;*
- *If $\mathbf{C}(p,\mathbf{a}) > 0$ then $\mathbf{Pre}(p,\mathbf{a}') = \mathbf{Pre}(p,\mathbf{a})$ and $\mathbf{C}(p,\mathbf{a}') = \omega$.*

Then $\mathbf{a}'$ is an acceleration.

Proof Consider $\{\sigma_n\}_{n\in\mathbb{N}}$ a family associated with the abstraction $\mathbf{a}$. We now show that the family $\{\sigma^n_n\}_{n\in\mathbb{N}}$ satisfies the conditions of Definition 6 for $\mathbf{a}'$. For all $n \in \mathbb{N}$:

- Let $p \in P$ such that $\mathbf{Pre}(p, \mathbf{a}') < \omega$.
  This implies that $\mathbf{C}(p, \mathbf{a}) \geq 0$ and that $\mathbf{Pre}(p, \mathbf{a}') = \mathbf{Pre}(p, \mathbf{a})$.
  Since, $\mathbf{C}(p, \sigma_n) \geq \mathbf{C}(p, \mathbf{a}) \geq 0$, one has
  $\mathbf{Pre}(p, \sigma_n^n) = \mathbf{Pre}(p, \sigma_n) \leq \mathbf{Pre}(p, \mathbf{a}) = \mathbf{Pre}(p, \mathbf{a}')$;
- Let $p \in P$ such that $\mathbf{C}(p, \mathbf{a}') = 0$. We get that $0 = \mathbf{C}(p, \mathbf{a}) \leq \mathbf{C}(\sigma_n)$.
  Therefore, $0 \leq n\mathbf{C}(\sigma_n) = \mathbf{C}(\sigma_n^n)$;
- Let $p \in P$ such that $\mathbf{Pre}(p, \mathbf{a}') < \omega$ and $\mathbf{C}(p, \mathbf{a}') = \omega$. This implies that $\mathbf{C}(p, \mathbf{a}) > 0$. So $1 \leq \mathbf{C}(p, \mathbf{a}) \leq \mathbf{C}(\sigma_n)$. Therefore, $n \leq n\mathbf{C}(\sigma_n) = \mathbf{C}(\sigma_n^n)$.

3 Karp and Miller's algorithm

Algorithm 1 is the Karp-Miller algorithm enlarged with 'ghost' variables (i.e. having no influence on the behavior of the algorithm) Acc and δ which will simplify the proof. Let us briefly describe this algorithm. It maintains a directed tree $Tr = (V, E, \lambda, \delta)$ whose vertices (V) are labeled by an ω-marking (function λ) and the arcs are labeled by a sequence of ω-transitions belonging to $T\mathsf{Acc}^*$, the function δ. We extend δ to a mapping from $E^* \mapsto (T\mathsf{Acc}^*)^*$ in the usual way. It maintains a subset of the vertices (Front) which are still to be explored. In order to shorten the description of the algorithm, we introduced $\mathsf{Anc}(u)$ the set of ancestors of u (excluding u).

As long as Front is not empty, the algorithm chooses a vertex $u \in \mathsf{Front}$. Then there are three cases:

- The marking of u is less then or equal to that of an ancestor u': then u is removed from Front and V and the edge entering u is removed.
- The marking of u is greater than that of an ancestor u' and for at least one place p, $\lambda(u')(p) < \lambda(u)(p) < \omega$. For all such places p, we substitute to its marking the value ω. *In our version*, we also define an ω-transition $\mathbf{a}$ by (1) defining it as the sequence of ω-transitions which labels the path from u' to u, (2) applying the transformation of Proposition 3, and (3) concatenating it with the sequence labeling the incoming arc of u.
- Otherwise the algorithm determine the fireable transitions and fire them to create the children of u which are inserted in Front. The vertex u is removed from Front. *In our version*, the incoming arc of a new vertex is labeled by the transition that has been fired.

When Front is empty, the algorithm ends. The set $Clover(\mathcal{N}, \mathbf{m}_0)$ corresponds to the maximal ω-markings associated with the vertices V.

Example 2 The Figure 2 illustrates the tree of Karp and Miller corresponding to the PN in Figure 1. Let us describe its construction during the development of the leftmost branch. From the initial marking, one fires t_1 which leads to $p_l + p_{bk}$, incomparable with $\mathbf{m}_0$. The exploration continues from this marking. The only fireable transition is t_5 whose firing leads to the marking $p_l + p_{bk} + p_{ba}$. An acceleration $\mathbf{a}_1$ is discovered with $\mathbf{Pre}(\mathbf{a}_1) = p_l$ and $\mathbf{C}(\mathbf{a}_1) = \omega p_{ba}$. The current marking is then modified accordingly to the firing of $\mathbf{a}_1$. This vertex is examined again during a subsequent iteration. There is no more acceleration possible. Consequently one continues the exploration: t_5 and t_6 are fireable. The vertex associated with the firing of t_5 has an identical marking: it will therefore be removed.

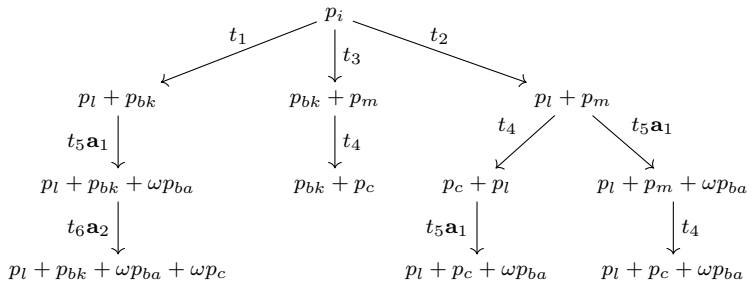

Fig. 2 Karp and Miller's tree

The vertex associated with firing of t_6 gives rise to a new acceleration: $\mathbf{a}_2$ with $\mathbf{Pre}(\mathbf{a}_2) = p_{bk} + \omega p_{ba}$ and $\mathbf{C}(\mathbf{a}_2) = \omega p_c + \omega p_{ba}$. Since from the last marking only t_5 and t_6 are fireable and lead to the same marking, the exploration of the branch is stopped. Note that $\mathbf{a}_1$ is re-discovered twice during the construction. The Cover computed here is represented by a set of 11 nodes and therefore 11 ω-markings. This set of 11 ω-markings is redundant but one can recover Clover by only keeping the 4 maximum elements: $\{p_i, p_l + p_{bk} + \omega p_{ba} + \omega p_c, p_{bk} + p_m, p_l + p_m + \omega p_{ba}\}$.

We will now show the correctness of the Karp and Miller algorithm, namely:

- It terminates;
- It is consistent: $\bigcup_{v \in V} [\![\lambda(v)]\!] \subseteq Cover(\mathcal{N}, \mathbf{m}_0)$;
- It is complete: $Cover(\mathcal{N}, \mathbf{m}_0) \subseteq \bigcup_{v \in V} [\![\lambda(v)]\!]$.

The termination is based on the fact that $\mathbb{N}_\omega^P$ is well ordered.

Proposition 4 (termination) *The Algorithm 1 terminates.*

Proof By contradiction, suppose the algorithm does not finish. A vertex of the tree can only be chosen in the loop at most $|P| + 1$ times. In fact, it remains in Front only if it has been accelerated, which implies that the associated marking has, at least, one more component which is equal to ω.
Consequently, the algorithm builds a tree whose number of vertices is infinite. Each vertex has at most $|T|$ children. By application of König's lemma, this tree has an infinite branch.
Let $\mathbf{m}_0, \mathbf{m}_1, \ldots$ be the markings associated with the vertices of this branch. $\mathbb{N}_\omega^P$ is well ordered. We can therefore extract an increasing infinite subsequence $\mathbf{m}_{\alpha(0)} \leq \mathbf{m}_{\alpha(1)} \leq \cdots$. There cannot be an equality between two consecutive vertices because then the second vertex would have been removed. We therefore deduce that an acceleration has been detected between any two consecutive vertices. Thus each marking has, at least, one more component which is equal to ω then its predecessor. Therefore this sequence contains at most $|P| + 1$ elements which contradicts the hypothesis.

The following lemma illustrates the advantage of the introduction of ghost variables.

Lemma 1 *For every edge $(u, v) \in E$, we have $\lambda(u) \xrightarrow{\delta(u,v)} \lambda(v)$.*

Algorithm 1: The Karp and Miller algorithm

```
KarpMiller(𝒩, m_0)
Input: A marked PN (𝒩, m_0)
Data: V a set of vertices; E ⊆ V × V; Front ⊆ V; λ : V → ℕ_ω^p; δ : E → TAcc*;
Tr = (V, E, λ, δ) a labeled tree; Acc a set of ω-transitions;
u, u', u'' vertices; a an ω-transition with non negative incidence;
Output: A labeled tree Tr = (V, E, λ, δ)
1  V ← {r}; E ← ∅; Front ← {r}; λ(r) ← m_0; Acc ← ∅;
2  while Front ≠ ∅ do
3      Choose u ∈ Front
4      if ∃u' ∈ Anc(u) s.t. λ(u') ≥ λ(u) then
5          Front ← Front \ {u} ; V ← V \ {u} ; E ← E \ V × {u} // λ(u) is covered
6      else if ∃u' ∈ Anc(u) s.t. λ(u') < λ(u) ∧ ∃p λ(u')(p) < λ(u)(p) < ω then
           // An acceleration is found between u and its ancestors u'
7          Let γ ∈ E* be the path from u' to u in Tr
8          a ← NewAcceleration()
9          foreach p ∈ P do
10             if λ(u')(p) < λ(u)(p) then λ(u)(p) ← ω
11             if C(p, δ(γ)) < 0 then Pre(p, a) ← ω; C(p, a) ← ω
12             if C(p, δ(γ)) = 0 then Pre(p, a) ← Pre(p, δ(γ)); C(p, a) ← 0
13             if C(p, δ(γ)) > 0 then Pre(p, a) ← Pre(p, δ(γ)); C(p, a) ← ω
14         end
15         Let (u'', u) be the incoming arc of u in Tr
16         δ((u'', u)) ← δ((u'', u)) · a; Acc ← Acc ∪ {a}
17     else
18         Front ← Front \ {u}
19         foreach t ∈ T s.t. λ(u) ≥ Pre(t) do
               // Adding the children of u
20             u' ← NewNode(); V ← V ∪ {u'}; Front ← Front ∪ {u'}; E ← E ∪ {(u, u')}
21             λ(u') ← λ(u) + C(t); δ((u, u')) ← t
22         end
23     end
24 end
25 return Tr
```

Proof There are two cases to be considered:

• **The creation of** (u, v)**.** This happens during the construction of the successors of u. Consequently, there exists a transition $t \in T$ such that $\lambda(u) \xrightarrow{t} \lambda(v)$ and this transition labels the edge (u, v) .

• **The modification of** $\lambda(v)$**.** This happens when the algorithm discovers an acceleration $\mathbf{a}$ between an ancestor u' of v and v. We denote by $\mathbf{m}^-$ the marking associated with v before its update and $\mathbf{m}^+$ the marking associated with v after its update. By induction, the sequence of ω-transitions along the path from u' to v is fireable from $\lambda(u')$, hence also from $\mathbf{m}^-$. This sequence has the same precondition that $\mathbf{a}$ has, except possibly on places p where $\mathbf{m}^-(p) = \omega$. So $\mathbf{a}$ is fireable from $\mathbf{m}^-$ and by construction $\mathbf{m}^- \xrightarrow{\mathbf{a}} \mathbf{m}^+$.

The following lemma is based on the preservation of abstractions by concatenation and the construction of accelerations from abstractions.

Lemma 2 *Every ω-transition $\mathbf{a} \in \mathsf{Acc}$ is an acceleration.*

Proof The proof is done by induction according to the order of insertion in Acc. Let $\mathbf{a} \in \mathsf{Acc}$ be a ω-transition. Let us denote by σ the sequence corresponding to

the path in the tree which led to the creation of $\mathbf{a}$. σ is a sequence of abstractions (by the induction hypothesis). From Proposition 2, this is an abstraction. The construction of $\mathbf{a}$ from σ corresponds to Proposition 3. $\mathbf{a}$ is therefore an acceleration.

The consistency of the algorithm is now a consequence of the previous lemmas.

Proposition 5 (consistency) *For all $v \in V$, $[\![\lambda(v)]\!] \subseteq Cover(\mathcal{N}, \mathbf{m}_0)$.*

Proof The proof is done by induction on the length of the path from r to u. The marking associated with r is $\mathbf{m}_0$. Hence $[\![\mathbf{m}_0]\!] \subseteq Cover(\mathcal{N}, \mathbf{m}_0)$. Denote by u the parent of v. By the induction hypothesis $[\![\lambda(u)]\!] \subseteq Cover(\mathcal{N}, \mathbf{m}_0)$. By Lemma 1, $\lambda(u) \xrightarrow{\delta(u,v)} \lambda(v)$. By Lemma 2, $\delta(u, v)$ is a sequence of abstractions. By Proposition 2, $\delta(u, v)$ is an abstraction. By Proposition 1, $[\![\lambda(v)]\!] \subseteq Cover(\mathcal{N}, \mathbf{m}_0)$.

In order to ease the proof of completeness, we introduce the notion of sequence of exploration, related to the coverability tree and its construction.

Definition 8 A sequence of transitions $\mathbf{m} \xrightarrow{\sigma} \mathbf{m}'$ is a *sequence of exploration* of Tr if there exists $v \in \mathsf{Front}$ with $\lambda(v) = \mathbf{m}$ and for all markings $\mathbf{m}''$ visited by the sequence and all $v \in V \setminus \mathsf{Front}$, we have : $\mathbf{m}'' \not\leq \lambda(v)$.

Lemma 3 *For all $\mathbf{m} \in Cover(\mathcal{N}, \mathbf{m}_0)$ at the start of each iteration of the main loop, the following holds:*

1. *Either there exists $v \in V \setminus \mathsf{Front}$ such that $\mathbf{m} \in [\![\lambda(v)]\!]$;*
2. *Or there exists a sequence of exploration $\mathbf{m}_1 \xrightarrow{\sigma} \mathbf{m}_2 \geq \mathbf{m}$.*

Proof We establish this result by induction on the number of iterations already performed.

• For all $\mathbf{m} \in Cover(\mathcal{N}, \mathbf{m}_0)$, there exists a sequence $\mathbf{m}_0 \xrightarrow{\sigma} \mathbf{m}_2 \geq \mathbf{m}$.
Initializing $V = \mathsf{Front} = \{r\}$ and $\lambda(r) = \mathbf{m}_0$. We get that Assertion 2 holds for the base case.

• Consider the start of an iteration of the loop. Pick $\mathbf{m} \in Cover(\mathcal{N}, \mathbf{m}_0)$. If $\mathbf{m}$ satisfies Assertion 1, it satisfies it until the termination of the algorithm.

Suppose that $\mathbf{m}$ satisfies Assertion 2. Let us denote the sequence of exploration $\mathbf{m}_1 \xrightarrow{\sigma} \mathbf{m}_2 \geq \mathbf{m}$ with $w \in \mathsf{Front}$ where $\lambda(w) = \mathbf{m}_1$. Let us consider the different alternatives.

∘ $\exists u' \in \mathsf{Anc}(u)$ such that $\lambda(u') \geq \lambda(u)$. This implies that $u \neq w$ and the sequence of exploration stays valid.

∘ $\lambda(u)$ is accelerated. If $u \neq w$, the sequence of exploration is still valid. If $u = w$ then, since $\lambda(u)$ has been increased, σ is fireable from $\lambda(u)$. Each marking visited is greater than or equal to the corresponding marking of the exploration sequence. So this new sequence is a sequence of exploration that covers $\mathbf{m}$.

∘ u is removed from Front and its children are computed. There are now two subcases to be considered.

– If for all marking $\mathbf{m}'$ visited by σ, $\mathbf{m}' \not\leq \lambda(u)$, the sequence of exploration is still valid.

– Otherwise, consider $\mathbf{m}'$ the last marking visited such that $\mathbf{m}' \leq \lambda(u)$ and the suffix of the sequence $\mathbf{m}' \xrightarrow{\sigma'} \mathbf{m}_2$.
If $\sigma' = \varepsilon$ then $\mathbf{m} \leq \mathbf{m}_2 = \mathbf{m}' \leq \lambda(u)$. Therefore Assertion 1 holds for $\mathbf{m}$.
Otherwise $\mathbf{m}' \xrightarrow{t} \mathbf{m}'' \xrightarrow{\sigma''} \mathbf{m}_2$. Since $\mathbf{m}' \leq \lambda(u)$, u has a child $v \in \mathsf{Front}$ such that $\lambda(u) \xrightarrow{t} \lambda(v) \geq \mathbf{m}''$. Therefore, $\lambda(v) \xrightarrow{\sigma''} \mathbf{m}^* \geq \mathbf{m}$ for some $\mathbf{m}^*$ and considering the choice of $\mathbf{m}'$ this sequence is a sequence of exploration.

Proposition 6 (completeness) *Upon termination of Algorithm 1,* $Cover(\mathcal{N}, \mathbf{m}_0) \subseteq \bigcup_{v \in V} [\![\lambda(v)]\!]$.

Proof When Algorithm 1 terminates, Front is empty. The completeness is a consequence of Lemma 3.

4 An improvement of the algorithm

In order to present an improvement of the algorithm, we deepen the study of accelerations. First, we equip the ω-transitions with an order related to their precondition and incidence.

Definition 9 Let P be a set of places and $\mathbf{a}$ and $\mathbf{a}'$ be ω-transitions.

$$\mathbf{a} \leq \mathbf{a}' \text{ if and only if } \mathbf{Pre}(\mathbf{a}) \leq \mathbf{Pre}(\mathbf{a}') \wedge \mathbf{C}(\mathbf{a}) \geq \mathbf{C}(\mathbf{a}')$$

In other words , $\mathbf{a} \leq \mathbf{a}'$ if $\mathbf{a}'$ is fireable from an ω-marking $\mathbf{m}$, then $\mathbf{a}$ is also firable and firing it leads to an ω-marking greater than or equal to the one reached by $\mathbf{a}'$.

Proposition 7 *Let $\mathcal{N}$ be a PN. Then the set of abstractions of $\mathcal{N}$ is upward closed. Similarly, the set of accelerations is upward closed in the set of ω-transitions with incidence in $\{0, \omega\}^P$.*

Proof Let $\mathbf{a}$ be an abstraction and $\mathbf{a}' \geq \mathbf{a}$ be an ω-transition. Let $\{\sigma_n\}_{n \in \mathbb{N}}$ be the family of sequences associated with $\mathbf{a}$. Let $n_0 = \max(\mathbf{C}(p, \mathbf{a}') \mid \mathbf{C}(p, \mathbf{a}') \in \mathbb{N})$ where, by convention, $\max(\emptyset) = 0$. We will show that the family $\{\sigma_{\max(n,n_0)}\}_{n \in \mathbb{N}}$ can be associated with $\mathbf{a}'$. Pick p such that $\mathbf{Pre}(p, \mathbf{a}') \in \mathbb{N}$, which implies that $\mathbf{Pre}(p, \mathbf{a}) \in \mathbb{N}$. We get:

– $\mathbf{Pre}(p, \sigma_{\max(n,n_0)}) \leq \mathbf{Pre}(p, \mathbf{a}) \leq \mathbf{Pre}(p, \mathbf{a}')$;
– If $\mathbf{C}(p, \mathbf{a}') \in \mathbb{Z}$ and $\mathbf{C}(p, \mathbf{a}) \in \mathbb{Z}$ then $\mathbf{C}(p, \sigma_{\max(n,n_0)}) \geq \mathbf{C}(p, \mathbf{a}) \geq \mathbf{C}(p, \mathbf{a}')$;
– If $\mathbf{C}(p, \mathbf{a}') \in \mathbb{Z}$ and $\mathbf{C}(p, \mathbf{a}) = \omega$ then $\mathbf{C}(p, \sigma_{\max(n,n_0)}) \geq n_0 \geq \mathbf{C}(p, \mathbf{a}')$;
– If $\mathbf{C}(p, \mathbf{a}') = \omega$ then $\mathbf{C}(p, \mathbf{a}) = \omega$ and $\mathbf{C}(p, \sigma_{\max(n,n_0)}) \geq n$.

The above also applies to accelerations.

Proposition 8 *Let $\mathcal{N}$ be a PN. Then the set of accelerations of $\mathcal{N}$ is well ordered.*

Proof The set of accelerations is a subset of $\mathbb{N}^P \times \{0, \omega\}^P$ with the order obtained by the Cartesian product of $(\mathbb{N}, \leq)$ and $(\{0, \omega\}, \geq)$. These sets are well ordered and since the Cartesian product preserves this property, the proposition follows.

Let us observe that the set of accelerations is not empty since it contains the acceleration $\mathbf{a}$ defined by $\mathbf{Pre}(\mathbf{a}) = \mathbf{C}(\mathbf{a}) = 0$ whose associated family $\{\sigma_n\}$ is defined by: for all n, $\sigma_n = \varepsilon$. Since the set of accelerations is well ordered and upward closed, it is equal to the upper closure of the finite set of *minimal* accelerations. We now study the maximal size of these minimal accelerations. Given a net, we denote by $d = |P|$ and $e = \max_{p,t}(\max(\mathbf{Pre}(p,t), \mathbf{Pre}(p,t) + \mathbf{C}(p,t))$.

We will use the following result by Jérôme Leroux (published on HAL in June 2019) which gives a bound to the length of the shortest transition sequences which connects two mutually reachable markings $\mathbf{m}_1$ and $\mathbf{m}_2$.

Theorem 1 *(Theorem 2, [15]) Let $\mathcal{N}$ be a PN, $\mathbf{m}_1, \mathbf{m}_2$ be markings, and σ_1, σ_2 be sequences such that $\mathbf{m}_1 \xrightarrow{\sigma_1} \mathbf{m}_2 \xrightarrow{\sigma_2} \mathbf{m}_1$. Then there exist σ'_1, σ'_2 such that $\mathbf{m}_1 \xrightarrow{\sigma'_1} \mathbf{m}_2 \xrightarrow{\sigma'_2} \mathbf{m}_1$ where:*

$$|\sigma'_1\sigma'_2| \leq ||\mathbf{m}_1 - \mathbf{m}_2||_\infty (3de)^{(d+1)^{2d+4}}$$

We deduce an upper bound on the size of minimal accelerations. Let $\mathbf{v} \in \mathbb{N}_\omega^P$. We denote by $||\mathbf{v}||_\infty = \max(\mathbf{v}(p) \mid \mathbf{v}(p) \in \mathbb{N})$.

Proposition 9 *Let $\mathcal{N}$ be a PN and $\mathbf{a}$ be a minimal acceleration.*
Then $||\mathbf{Pre}(\mathbf{a})||_\infty \leq e(3de)^{(d+1)^{2d+4}}$.

Proof We consider the net $\mathcal{N}' = \langle P', T', \mathbf{Pre}', \mathbf{C}' \rangle$ obtained from $\mathcal{N}$ by removing the set of places $\{p \mid \mathbf{Pre}(p, \mathbf{a}) = \omega\}$ and adding the set of transitions $T_1 = \{t_p \mid p \in P'\}$ with $\mathbf{Pre}(t_p) = p$ and $\mathbf{C}(t_p) = -p$.
We denote $P_1 = \{p \mid \mathbf{Pre}(p, \mathbf{a}) < \omega = \mathbf{C}(p, \mathbf{a})\}$. Let $\mathbf{m}_1$ the marking obtained by restricting $\mathbf{Pre}(\mathbf{a})$ to P' and $\mathbf{m}_2 = \mathbf{m}_1 + \sum_{p \in P_1} p$. Observe that $d' \leq d$ and that $e' = e$.
Let $\{\sigma_n\}_{n \in \mathbb{N}}$ be the family of sequences associated with $\mathbf{a}$.
Consider $n^* = ||\mathbf{Pre}(\mathbf{a})||_\infty + 1$. Then σ_{n^*} is fireable in $\mathcal{N}'$ from $\mathbf{m}_1$ and its firing reaches a marking that covers $\mathbf{m}_2$. By concatenating transitions of T_1, we obtain a firing sequence in $\mathcal{N}'$ such that $\mathbf{m}_1 \xrightarrow{\sigma_1} \mathbf{m}_2$. By the same process, we obtain a sequence $\mathbf{m}_2 \xrightarrow{\sigma_2} \mathbf{m}_1$.
Applying Theorem 1, there exists a sequence σ'_1 with $\mathbf{m}_1 \xrightarrow{\sigma'_1} \mathbf{m}_2$ and $|\sigma'_1| \leq (3de)^{(d+1)^{2d+4}}$ since $||\mathbf{m}_1 - \mathbf{m}_2||_\infty = 1$. By deleting transitions of T_1 in σ'_1, we obtain a sequence $\sigma''_1 \in T^*$ with $\mathbf{m}_1 \xrightarrow{\sigma''_1} \mathbf{m}'_2 \geq \mathbf{m}_2$ and $|\sigma''_1| \leq (3de)^{(d+1)^{2d+4}}$.
The ω-transition $\mathbf{a}'$, defined by:
- $\mathbf{Pre}(p, \mathbf{a}') = \mathbf{Pre}(p, \sigma''_1)$ for all $p \in P'$;
- $\mathbf{Pre}(p, \mathbf{a}') = \omega$ for all $p \in P \setminus P'$;
- and $\mathbf{C}(\mathbf{a}') = \mathbf{C}(\mathbf{a})$.

is an acceleration with associated family $\{\sigma''^n_1\}_{n \in \mathbb{N}}$.
By definition of $\mathbf{m}_1$, $\mathbf{a}' \leq \mathbf{a}$. Since $\mathbf{a}$ is minimal, $\mathbf{a}' = \mathbf{a}$.
Since $|\sigma''_1| \leq (3de)^{(d+1)^{2d+4}}$, $||\mathbf{Pre}(\mathbf{a})||_\infty = ||\mathbf{Pre}(\mathbf{a}')||_\infty \leq e(3de)^{(d+1)^{2d+4}}$.

Proposition 10 *Let $\mathcal{N}$ be a PN and $\mathbf{a}$ be an acceleration.*
Then the ω-transition $trunc(\mathbf{a})$ defined by:
- $\mathbf{C}(trunc(\mathbf{a})) = \mathbf{C}(\mathbf{a})$;

– *for all* p *such that* $\mathbf{Pre}(p, \mathbf{a}) \neq \omega$,
$\mathbf{Pre}(p, trunc(\mathbf{a})) = \min(\mathbf{Pre}(p, \mathbf{a}), e(3de)^{(d+1)^{2d+4}})$;
– *for all* p *such that* $\mathbf{Pre}(p, \mathbf{a}) = \omega$, $\mathbf{Pre}(p, trunc(\mathbf{a})) = \omega$.
is an acceleration.

Proof Let $\mathbf{a}' \leq \mathbf{a}$, be a minimal acceleration. For all p such that $\mathbf{Pre}(p, \mathbf{a}) \neq \omega$, $\mathbf{Pre}(p, \mathbf{a}') \leq e(3de)^{(d+1)^{2d+4}}$. Hence $\mathbf{a}' \leq trunc(\mathbf{a})$. Since the set of accelerations is upward closed, we deduce that $trunc(\mathbf{a})$ is an acceleration.

Algorithm 2: An acceleration of the Karp and Miller algorithm

```
   KarpMillerImproved(𝒩, m_0)
   Input: A marked PN (𝒩, m_0)
   Data: V set of vertices; E ⊆ V × V; Front ⊆ V; λ : V → ℕ_ω^p; δ : E → T Acc*;
   Tr = (V, E, λ, δ) a labeled tree; Acc a set of ω-transitions;
   u, u', u'' vertices; a an acceleration;
   Output: A labeled tree Tr = (V, E, λ, δ)
 1 V ← {r}; E ← ∅; Front ← {r}; λ(r) ← m_0; Acc ← ∅;
 2 while Front ≠ ∅ do
 3     Choose u ∈ Front and let u'' be the predecessor of u
 4     foreach a ∈ Acc s.t. λ(u) -a-> λ(u) + C(a) > λ(u) do
 5         λ(u) ← λ(u) + C(a); δ((u'', u)) ← δ((u'', u))a
 6     end
 7     if ∃u' ∈ Anc(u) s.t. λ(u') ≥ λ(u) then
 8         Front ← Front \ {u} ; V ← V \ {u} ; E ← E \ V × {u} // λ(u) is covered
 9     else if ∃u' ∈ Anc(u) s.t. λ(u') < λ(u) ∧ ∃p λ(u')(p) < λ(u)(p) < ω then
           // An acceleration is found between u and an ancestors of u
10         Let γ ∈ E* The path from u' to u in Tr
11         a ← NewAcceleration()
12         foreach p ∈ P do
13             if C(p, δ(γ)) < 0 then Pre(p, a) ← ω; C(p, a) ← ω
14             if C(p, δ(γ)) = 0 then Pre(p, a) ← Pre(p, δ(γ)); C(p, a) ← 0
15             if C(p, δ(γ)) > 0 then Pre(p, a) ← Pre(p, δ(γ)); C(p, a) ← ω;
                 λ(u)(p) ← ω
16         end
17         a ← trunc(a)
18         δ((u'', u)) ← δ((u'', u)) · a; Acc ← Acc ∪ {a}
19     else
20         Front ← Front \ {u}
21         foreach t ∈ T s.t. λ(u) ≥ Pre(t) do
               // Adding the children of u
22             u' ← NewNode(); V ← V ∪ {u'}; Front ← Front ∪ {u'}; E ← E ∪ {(u, u')}
23             λ(u') ← λ(u) + C(t); δ((u, u')) ← t
24         end
25     end
26 end
27 return Tr
```

We are now able to describe the improvement made to the construction of Karp and Miller (see Algorithm 2). First when one discovers an acceleration, one truncates it before inserting it into Acc (line 17). Then, when a vertex of the Front is selected, one first tries to apply the accelerations of Acc to increase its marking (lines 4-6).

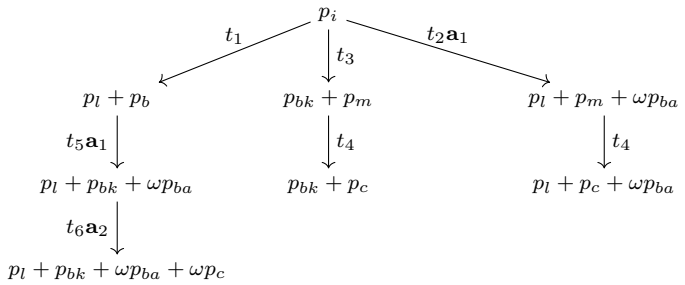

Fig. 3 An accelerated tree

Example 3 Figure 3 shows the accelerated Karp and Miller tree corresponding to the net of Figure 1. When the vertex obtained by firing t_2 from the initial marking is examined, the algorithm evaluates whether firing $\mathbf{a}_1$ or $\mathbf{a}_2$ (both discovered in the left branch) is possible. The acceleration $\mathbf{a}_1$ is fireable and therefore fired.

The proof of termination is unchanged while the proofs of consistency and completeness require only very minor modifications to integrate the case of the application of accelerations. The the proof of consistency remains valid since the $\omega-$transition $\mathbf{a}$ which is truncated before being added to Acc is still an acceleration due to Proposition 10. For completeness one needs to include in the induction step of the proof of Lemma 3, the case of the algorithm using a previously discovered acceleration.

In practice applying the memorized accelerations decreases the size of tree. Furthermore the cost of memorizing accelerations is largely compensated by this decrease.

From a theoretical point of view, there is at most a doubly exponential number of accelerations each of exponential size: that is to say an additional doubly exponential memory complexity. Recall, that the size of a cover tree is in the worst case non-primitive recursive. Therefore even without decrease of the size of the tree, the increase of memory size is negligible. Moreover if the memory space is a strong constraint then it is enough to keep a subset of the accelerations since the proof of the modified algorithm is valid for any set of accelerations.

5 An efficient tool for the clover construction

Based on the improvements discussed in the previous section and the algorithm (from now on denoted by `AF`) from [6], we have designed an algorithm for computing the minimal coverability set. The main idea of this improvement consists in avoiding the redundancy between ω-markings created by the Karp and Miller algorithm. Hence the new algorithm refines algorithm 2 by adding the following steps to the main loop after using previous discovered accelerations (Line 6):

1. **Cleaning**: If there exists a vertex with an ω-marking greater than the one of the current processed vertex, then the current vertex is redundant, the algorithm deletes it and starts an new iteration of the main loop.

2. **Pruning**: Before computing the successors of the current vertex, the algorithm prunes all the subtrees whose roots have ω-markings smaller than the one of the currently processed vertex.

Observe that the main difference between this algorithm and `AF` is the reuse of previously discovered acceleration which transforms an incomplete algorithm (see [7]) into a complete one. For further details and the whole proof of the correctness, we refer the reader to [9].

This algorithm is implemented in the tool `MinCov`, a tool designed to efficiently solve the coverability and minimal coverability set problems. `MinCov` is implemented in Python 3.7 using the Numpy and the Z3-solver libraries, and it is around 2000 lines of code. `MinCov` imports Petri net in .spec format from Mist[1]. `MinCov` and the benchmarks discussed below can be found here: `https://github.com/IgorKhm/MinCov`.

Benchmarks. We split the benchmarks into two sets, a set of benchmarks from the literature, and a set of randomly generated Petri nets. The literature ones were taken from [1,4] which in turn were gathered from five sources:

- **MIST**[1]: containing both real and artificial systems (mutual exclusion protocols, communication protocols, ...);
- **BFC**[2]**[12]**: systems originated from analysis of concurrent C programs;
- **SOTER**[3]**[3]**: systems originating from the analysis of Erlang programs in order to test the tool SOTER;
- **Medical[14]**: systems originating from the analysis of a simple medical messaging system of Vanderbilt University Medical Center;
- **Bug tracking[14]**: systems originating from the analysis of messages of a bug-tracking system.

The random Petri nets were generated by `MinCov` with the following properties: (1) $50 < |P|, |T| < 100$, (2) the number of places connected to each transition is bounded by 10, and (3) they are not structurally bounded.

We compare `MinCov` with the tool `MP` [17], the tool `VH` [18] and the tool `CovProc` [10]. We have also implemented the (incomplete) minimal coverability tree algorithm `AF`, in order to measure the additional memory needed for the (complete) tools. Both `MP` and `VH` tools were sent to us by the courtesy of the authors. The tool `MP` has two implementations one in Python and the other in C++, for this comparison we took the Python one, in order to compare both tools implemented in the same language. The tool `CovProc` participates only in the benchmark from the literature, since it is extremely slow, and could not achieve the analysis of most of randomly generated Petri nets in a reasonable time. All the benchmarks were performed on a single computer equipped with Intel i5-8250U CPU with 4 cores, 16 GB of memory and Ubuntu Linux 18.03.

The execution time of the tools was limited to 900 seconds for each Petri net, and the results are summarized in the tables below. Each of these tables contain the following information: The first column (T/O) shows the number of instances on which the tool timed out. The second column (Time) consists of the total time on instances that did not time out plus 900 seconds for any instance that led to a

[1] `https://github.com/pierreganty/mist/wiki`.

[2] `http://www.cprover.org/bfc/`.

[3] `http://mjolnir.cs.ox.ac.uk/soter/`

time out. The third column (#Nodes) consists of the sum of peak number of nodes in instances that did not time out on any of the tools (except `CovProc` which does not provide this number). For `MinCov` we consider the peak number of nodes plus the number of accelerations.

123 benchmarks from the literature

	T/O	Time	#Nodes
`MinCov`	16	18127	**48218**
`VH`	**15**	**14873**	75225
`MP`	24	23904	478681
`CovProc`	49	47081	N/A
`AF`	19	19223	45660

1078 random benchmarks

	T/O	Time	#Nodes
`MinCov`	146	159940	**1291066**
`VH`	**111**	**110431**	2454490
`MP`	231	260608	31354531
`CovProc`	N/A	N/A	N/A
`AF`	163	178322	1267076

Table 1 Benchmarks for `MinCov`.

In the benchmarks from the literature we observed that the instances that timed out from `MinCov` are included in those of `AF` and `MP`. However there were instances the timed out on `VH` but did not time out on `MinCov` and vice versa. W.r.t. memory requirements `AF` and `MinCov` have the least number of nodes. `MinCov` is the second fastest tool, and compared to `VH` it is 1.2 times slower on the examples from the literature. A possible explanation would be that `VH` is implemented in C++.

In the random benchmarks we observe the same behavior as in the one from the literature where `MinCov` is only second time-wise, it is 1.5 times slower then `VH`, but it is the most efficient memory-wise. Compared to `AF` the extra memory used by `MinCov` is negligible: 0.002 times additional nodes in average.

6 Conclusion

The study of the Karp and Miller algorithm led us to two results. First we have developed a simple and elegant proof of this algorithm based on the new notions of abstraction, acceleration and exploration sequence. Then we designed an accelerated version of this algorithm which memorizes all the accelerations already computed in order to re-apply them systematically.

We have also implemented an accelerated version of the minimal coverability graph construction. We have compared the performance of this algorithm with the main existing algorithms calculating Clover with promising results.

In the future, we will also study the possibility of effectively pre-calculating the set of minimal accelerations or some relevant subset. Finally it would be interesting to introduce and apply the concept of acceleration for the study of other well structured transition systems.

Acknowledgments. The authors would like to thank the reviewers for their careful reading of the article and their constructive comments that help us to improve the quality of our article.

References

1. Michael Blondin, Alain Finkel, Christoph Haase, and Serge Haddad. The logical view on continuous Petri nets. *ACM Transactions on Computational Logic (TOCL)*, 18(3):24:1–24:28, 2017.
2. Michael Blondin, Alain Finkel, and Pierre McKenzie. Well behaved transition systems. *LMCS*, 13(3):1–19, September 2017.
3. Emanuele D'Osualdo, Jonathan Kochems, and C. H. Luke Ong. Automatic verification of Erlang-style concurrency. In *Static Analysis*, pages 454–476, 2013.
4. Javier Esparza, Ruslán Ledesma-Garza, Rupak Majumdar, Philipp Meyer, and Filip Niksic. An smt-based approach to coverability analysis. In *CAV*, pages 603–619, 2014.
5. Alain Finkel. Reduction and covering of infinite reachability trees. *Information and Computation*, 89(2):144–179, 1990.
6. Alain Finkel. The minimal coverability graph for Petri nets. In *Advances in Petri Nets 1993*, volume 674 of *Lecture Notes in Computer Science*, pages 210–243. Springer, 1993.
7. Alain Finkel, Gilles Geeraerts, Jean-François Raskin, and Laurent Van Begin. A counter-example the the minimal coverability tree algorithm. Technical Report 535, Université Libre de Bruxelles, Belgium, 2005.
8. Alain Finkel and Jean Goubault-Larrecq. Forward analysis for WSTS, part II: Complete WSTS. *LMCS*, 8(4), 2012.
9. Alain Finkel, Serge Haddad, and Igor Khmelnitsky. Minimal Coverability Tree Construction Made Complete and Efficient. In *https://hal.inria.fr/hal-02479879/*, 2020.
10. Gilles Geeraerts, Jean-François Raskin, and Laurent Van Begin. On the efficient computation of the minimal coverability set of Petri nets. *Int. J. Foundations of Computer Science*, 21(2):135–165, 2010.
11. Michel Hack. *Decidability questions for Petri Nets.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1976.
12. Alexander Kaiser, Daniel Kroening, and Thomas Wahl. A widening approach to multi-threaded program verification. *ACM Trans. Program. Lang. Syst.*, 36(4), 2014.
13. Richard M. Karp and Raymond E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(2):147–195, 1969.
14. Johannes Kloos, Rupak Majumdar, Filip Niksic, and Ruzica Piskac. Incremental, inductive coverability. In *CAV*, pages 158–173, 2013.
15. Jérôme Leroux. Distance between mutually reachable petri net configurations. In *FSTTCS 2019*, pages 47:1–47:14, 2019.
16. Artturi Piipponen and Antti Valmari. Constructing minimal coverability sets. *Fundamenta Informaticae*, 143(3–4):393–414, 2016.
17. Pierre-Alain Reynier and Frédéric Servais. Minimal coverability set for Petri nets: Karp and Miller algorithm with pruning. *Fundamenta Informaticae*, 122(1–2):1–30, 2013.
18. Antti Valmari and Henri Hansen. Old and new algorithms for minimal coverability sets. *Fundamenta Informaticae*, 131(1):1–25, 2014.
19. Mitsuharu Yamamoto, Shogo Sekine, and Saki Matsumoto. Formalization of karp-miller tree construction on petri nets. In *Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs*, CPP 2017, pages 66–78, New York, NY, USA, 2017. ACM.